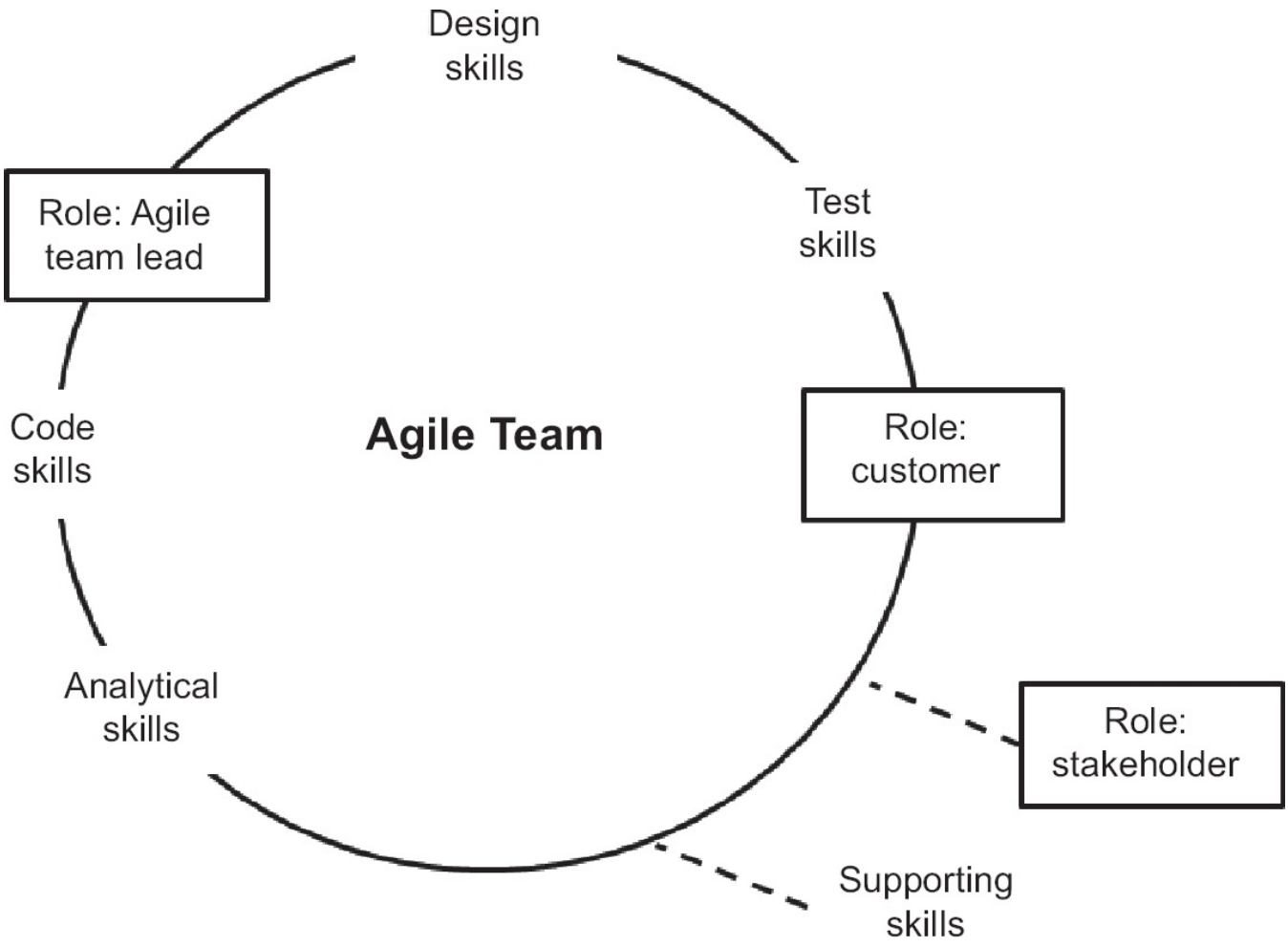
6 COMMON AGILE ROLES

Each Agile framework (see Chapter 14) has its own view on the different roles in an Agile delivery, with their own names and structure. The roles that are discussed in this chapter occur generically across the majority of Agile frameworks focused on the team level; there are other roles, such as Agile project manager, that would typically be applied when delivering Agile projects (see Section 14.4) or scaling Agile (see Section 14.8), but will not be covered here.

There are generally four key roles in an Agile delivery: the customer, the Agile lead, the Agile team itself and stakeholders. Figure 6.1 highlights the typical skills that will be required in an Agile team; however, an Agile approach aims for team members to be 'specialised generalists' (see Section 6.2.2) rather than just having one skill. The dotted lines represent that there may be many other stakeholders and many other supporting skills that contribute to the team but may not be part of the core team (e.g. architect skills).

Figure 6.1 Generic Agile roles



In the following sections, we will look at the four roles in more detail.

6.1 THE CUSTOMER

'Customer' in an Agile context means the person who makes the decisions about what will be done in what order, and the term normally refers to the owner of the

product being built. The customer has to be able to answer the question ‘Why are we doing this?’ – essentially they are responsible for the ‘vision’ of the product.

The customer also

- defines what stories are to be delivered;
- decides the order of the stories on the backlog;
- signs off stories as ‘done’ (see Section 10.2) at the end of each iteration/sprint or release.

A good and regularly used mnemonic for remembering the key criteria of a ‘good customer’ is ‘DARK(a)’, which stands for…

D is for Desire A customer should want to be involved in and excited about the product being delivered.

A is for Authority The customer should have the authority to make decisions and enforce them. It must be clear to the customer, team and stakeholders what decisions the customer can and cannot make. For example, the customer working with a team may be part of a larger project of multiple teams and therefore they would not have the authority to make decisions concerning the larger project. For the decisions the customer cannot make it must be clear who can make those decisions and what the turnaround time will be; this is key to enabling planning.

R is for Responsibility The customer is (amongst other things) responsible for defining what the team will deliver and in which order.

K is for Knowledgeable This does not mean that the customer has to be the font of all knowledge but it does mean that they must know where to find the information in a timely manner.

A is for Availability When working in small, focused cycles, it is vital that any decisions are made in the same timely manner and any questions are answered as quickly as possible. This cannot happen if the customer is absent. This does not necessarily mean that the customer needs to be co-located and available 24/7. However, the role is responsible for the day-to-day communication channels between the team and the business.

The role and responsibilities of the customer are also covered in the following Agile Manifesto statements and principles.

6.1.1 Customer collaboration over contract negotiation

There is value in defining a contract in an Agile delivery; however, the contract is more about stating clearly the collaboration between the customer and the team, rather than focusing on up-front analysis and design. The contract will identify the customer, their agreed level of authority, who will make any decisions they cannot make, and in what timescale these decisions will need to be made.

6.1.2 Our highest priority is to satisfy the customer through early

and continuous delivery of valuable software

An Agile delivery focuses on delivering value to the customer as early as possible and in a continuous manner. To enable this, the customer needs to define the stories that they require, together with the value associated with those stories and the order in which they want them delivered. The customer is responsible for creating stories in collaboration with stakeholders and the team.

Arguably the key word in this manifesto principle is ‘valuable’. If for some reason the team (for example) have added stories to the backlog on their own and the customer does not understand the stories, collaboration becomes difficult and risk is introduced. Therefore it is essential that the way the stories are described makes sense to the customer and uses non-technical language and as little jargon as possible.

6.1.3 Welcome changing requirements, even late in development. Agile processes harness change for the customer’s competitive advantage

Agile designs emerge as the product is being developed (see Section 9.2). This means that any changed or new requirement that gives the customer added capability to deliver competitive advantage quickly and continuously can be added into the delivery based on comparative business value.

In a time-boxed delivery, if the stories to be delivered change right at the end of the project or release, then neither Waterfall nor Agile will save the project. However, in Agile this scenario is extremely rare because the team delivers stories continuously in short iterations/sprints, and continually interacts with all stakeholders, demonstrating to them the working software that has been developed. Therefore it will be extremely unusual that fundamental requirement changes occur right at the end of a release or project.

6.1.4 Business people and developers must work together daily throughout the project

To enable continuous delivery of valuable product, business people (i.e. stakeholders and customers) and the team must work together throughout the project. If this does not happen, the product that is developed is highly unlikely to be suitable for the customer.

6.1.5 The most efficient and effective method of conveying information to and within a development team is face-to-face conversation

The role of the customer in the team is to enable accurate communication and fast feedback cycles. To effectively convey information, the customer and team must have the ability to communicate face-to-face, whether that is physically or virtually (see Section 8.2).

6.2 THE TEAM

Agile teams are typically responsible for:

- Deciding how to do the work. This involves refining stories with the customer and then (typically) dividing them into tasks.
- Defining how long the work will take. With the detailed task breakdown, the team will be able to provide a more accurate estimate of the effort needed to complete each task. Typically, tasks will be broken down to be completed in less than 10 hours (2 days) of effort.
- Deciding who does the work. During the daily stand-up meeting (see Section 8.3), the team will have an opportunity to examine the progress achieved and decide how to best organise themselves to deliver maximum value to the customer.
- Delivering the product.

As we've seen in Section 6.1 the customer is responsible for defining what stories will be developed in what order, relying on support from the team. At iteration/sprint planning the team will then subdivide these stories into tasks, decide who will deliver the tasks and how long (typically in hours) delivery will take. Very experienced teams may not plan at task level as their level of skill and experience enables them to deliver effectively without that level of planning. However, it is generally a mistake to not go down to the task level in planning, as this is a very common cause of sprint/iteration goals being missed.

6.2.1 Specialised generalists

Agile teams should consist of members who are 'specialised generalists'. A specialised generalist is someone who does have a specialist skill (like testing skill) as well as a general understanding of what everyone else in the team does. For example, in a football team each member would have a specialist skill (e.g. goalkeeper, centre forward), but they would also have general skills such as the ability to tackle, kick the ball and so on.

Having a team made up of only specialists can make communication and therefore collaboration and teamwork very difficult as the understanding of what everyone else in the team is doing can be missing. Typical specialist skills that can be found in an Agile team are:

- architectural skill;
- analytical skill;
- design skill;
- coding skill;
- testing skill;
- business knowledge.

6.2.2 Self-organising teams

Agile teams are generally set up as self-organising teams, meaning they have the authority to decide how work gets done, who does it and how long it should take. This does not mean that they have carte blanche to do whatever they like. Teams will typically be operating within a structure of high-level standards and guidelines that ensure consistency across a programme, project or organisation. Part of the skill of the Agile lead (see Section 6.3) is to decide where to put these constraints for the good of the overall organisation and the good of the team, without adversely affecting self-organisation.

Most self-organising teams, for example, would not have the authority to choose who is on the team, what business problem they are expected to help solve, or even what budget they have. Even from a technical perspective, most teams would not have a free choice to use any architecture or toolset they think was appropriate as these limits are often imposed by external enterprise or system architects.

Being part of a self-organising team allows team members to learn and adjust constantly as they develop the product, which in turn leads to better architectures, requirements and designs. This principle is based on the understanding that software development is knowledge-based work, and a knowledge worker is somebody who knows more about their job than their boss (Drucker, 2001).

Table 6.1 shows examples of behaviours from self-organising teams and teams that are managed by command and control.

Table 6.1 Typical differences between team types

Command-and-control teams	Self-organised teams
Take directions	Take initiative
Seek individual reward	Focus on team contributions
Focus on low-level objectives	Concentrate on solutions
Compete	Cooperate
Comply with processes regardless of outcome	Continually look for better ways of working
React to emergencies	Take steps to prevent emergencies

Other reasons why self-organising teams tend towards better performance for software development work (30–50% or more (Orsburn, 1990)) are:

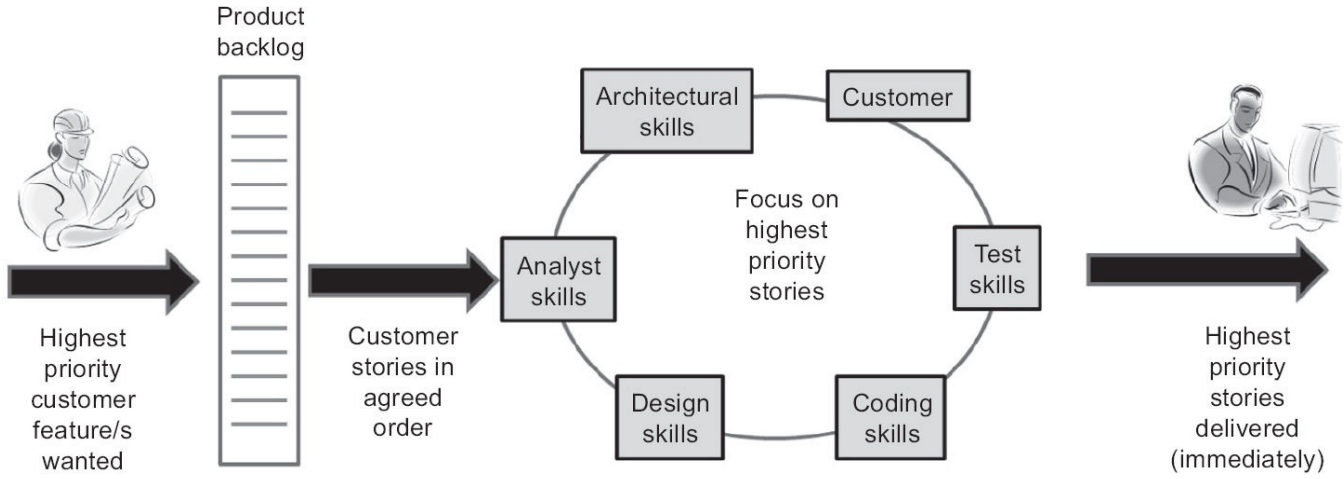
- **Fast decision making** – many decisions don’t need to go up the management chain.
- **Increased motivation** – due to a greater sense of autonomy.
- **Increased brain power** – instead of a single manager working on a problem, there is the potential to apply the brain power of the entire team.

- **Increased levels of initiative and continuous improvement** – to the point where team members start to consider the goals of the team to be more important than their own personal goals.

6.2.3 Feature teams

An Agile feature team concentrates on producing ordered stories, refined as value-add features in the value sequence that the customer requires (see Figure 6.2). It contains whatever skills are required to deliver the stories to the customer and is a key driver in the reduction of 'waste' in the value chain (see Chapter 13).

Figure 6.2 The Agile feature team



Agile teams are usually structured as feature teams as it enables delivery of features (described as stories) from the backlog in vertical slices (see Section 13.2). This allows for regular delivery of value-add features to the customer.

6.2.4 Component teams

Sometimes – in particular when Agile is scaled across large complex deliveries – it is not realistic for all teams to be feature teams as it may only be possible to deliver features to the customer once a number of components have been delivered and integrated. This is when component teams are required. Component teams are responsible for producing components that can be combined with other components from other teams and integrated together into a feature.

So for example, the production of a mobile phone operating system would require many Agile teams, some of them component teams, as it would be very difficult for a single team to deliver features that a phone user would understand within weeks or months.

6.2.5 Core and support teams

In large and complex deliveries, it is quite typical to find core teams that are focused on delivering across the value chain, as well as some support teams (e.g. possibly architecture support or user interface support). A typical size of a core team is somewhere between three and nine people, not including the customer, the Agile lead or support people.

Support teams exist specifically to support and enable the core teams to do things right first time; not to 'police' the core teams and tell them when they have done something wrong at a delivery gateway.

6.3 THE AGILE LEAD

The role of an Agile lead is a multifaceted one. It is named and described differently in different Agile frameworks, but the Agile lead's main responsibility across all frameworks is to enable their team to self-organise and continually improve. Some Agile frameworks see the Agile lead role as a change agent; others describe the role to include more team leadership responsibilities in addition to the change agent capabilities.

An Agile lead should have the following characteristics:

- experience and maturity to understand that the focus in Agile is on the individuals and the effective face-to-face interactions between those individuals;
- the ability to create a no-blame culture where people are not afraid to try new things, and an understanding that everyone (including themselves) makes mistakes, and that mistakes are acceptable as long as learning occurs;
- experience in and knowledge of Agile;
- experience in transformation.

The Agile lead should also have an appreciation of the aspects detailed in the following subsections.

6.3.1 Agile lead as 'servant-leader'

The term 'servant-leader' was termed in 1970 in an essay by Robert K. Greenleaf:

> The Servant-Leader is servant first… It begins with the natural feeling that one wants to serve, to serve first. Then conscious choice brings one to aspire to lead. That person is sharply different from one who is leader first; perhaps because of the need to assuage an unusual power drive or to acquire material possessions.
>
> (Greenleaf, 1970)

An Agile lead should be a servant-leader who gains the credibility in their team by being a servant to the team's needs and wants. They will not command and control the team; instead they will facilitate and enable the team to self-organise.

In contrast, a team lead who operates in a 'command and control' style will typically pass a plan to the team that details how they should do things (tasks) and how long they have to do those things; they will then control the team to achieve the plan. If the team fails, which is likely because they will not buy into a plan they had little or no input to, the team will probably blame the team lead for the failure citing excuses such as 'we knew the plan was wrong'. The team will also not self-organise as they have no motivation for doing so and they won't feel safe to do so.

6.3.2 Removing impediments

An Agile lead is also responsible for supporting the team in identifying and finding ways of removing anything that blocks a task from being executed or a feature from being delivered. Impediments might include:

- management command and control behaviours;
- little support from management for Agile way of working;
- implementing Agile within a Waterfall management environment;
- ineffective transformation – nothing ever changes;
- technology that blocks Agile capability;
- lack of empowerment;
- lack of ability to self-organise;
- organisation not structured to enable agility.

Sometimes impediments can also simply be interruptions that the team may experience, such as being continuously asked to 'just do this or that'. All interruptions create noise that can impede the team, and therefore hinder delivery. The Agile lead should remove as much noise from the team as possible.

6.3.3 Workshop facilitator

Many Agile frameworks describe activities that are most effectively delivered as workshops, for example 'show and tells' (see Section 8.4). All of these activities and any other ad-hoc workshops need to be structured and facilitated, and it is common for Agile leads to take on this role.

As a facilitator should, if possible, be independent from the people who attend the workshop, it may be a good idea for the lead from team A to facilitate workshops for team B and vice versa.

6.3.4 Process facilitator

Another responsibility of Agile leads is to help the organisation, programme, project or team to define an Agile operating model (see Section 5.1). So for example, Agile leads need to make sure the team's definition of 'done' or process policies are being met (see Section 10.2.1), that work in progress limits are being adhered to, that all Agile activities are respected, and that stories are refined to the appropriate size.

Agile leads should also guide their team to identify areas of their processes that they can improve and should introduce process improvements by facilitating the self-organising team rather than by imposing them.

6.3.5 Coach/trainer

Agile leads should pass on their knowledge of Agile principles, processes and practices to their team through coaching and training. Transformation starts with

visualisation – if the group to be transformed cannot visualise why they should transform, they won't!

Agile leads will often champion the implementation of new technical practices. Agile adoption does not always mandate the need for new technical practices and tools, but they are often good enablers for a team to become more Agile. This means that Agile leads need to have a good understanding of technical practices; eXtreme Programming (see Section 14.1) is a good place to start the journey to this level of understanding.

Implementation of new practices needs to be done with the buy-in of the team, following the principles of self-organisation. If new practices are imposed upon the team by the Agile lead without the team buying in to them, they are likely to fail.

6.4 THE STAKEHOLDERS

The term stakeholder refers to all the people and organisations that have a real or perceived 'stake' in the project or its outcomes. PRINCE2 defines a stakeholder as:

> Parties with an interest in the execution and outcome of a project. They would include business streams affected by or dependent on the outcome.
>
> (PRINCE2, 2011)

The *PMBOK® Guide* (*A Guide to the Project Management Body of Knowledge*) (PMI, 2013) describes a stakeholder as a person or organisation that:

- is actively involved in the project;
- has interests that may be positively or negatively affected by the performance or completion of the project;
- may exert influence over the project, its deliverables or its team members.

Another definition may be: 'Any person or group who can help us, or hinder.'

Identifying and effectively managing stakeholders is key to any delivery's success, as they can champion a project and help drive success, or be very effective saboteurs. Powerful stakeholders are much more likely to sabotage a project if they don't feel engaged.

> Many projects fail to involve one or more critically important stakeholders during project definition and planning. The resulting problems are easily predictable: requirements conflicts and rework, at a minimum; and sometimes more dire consequences, including lawsuits or hefty fines.
>
> (Verzuh, 2008)

The role of the stakeholder in an Agile delivery is to ensure that the interests of the group they are part of are represented. In the majority of Agile frameworks the customer defines what the team will do in what order by interacting with the stakeholders. However, direct interaction between the team and the stakeholders is also encouraged as it enables the team to ascertain key information they need to deliver an effective product.

6.4.1 Identifying stakeholders

There are a number of techniques to identify effective stakeholders:

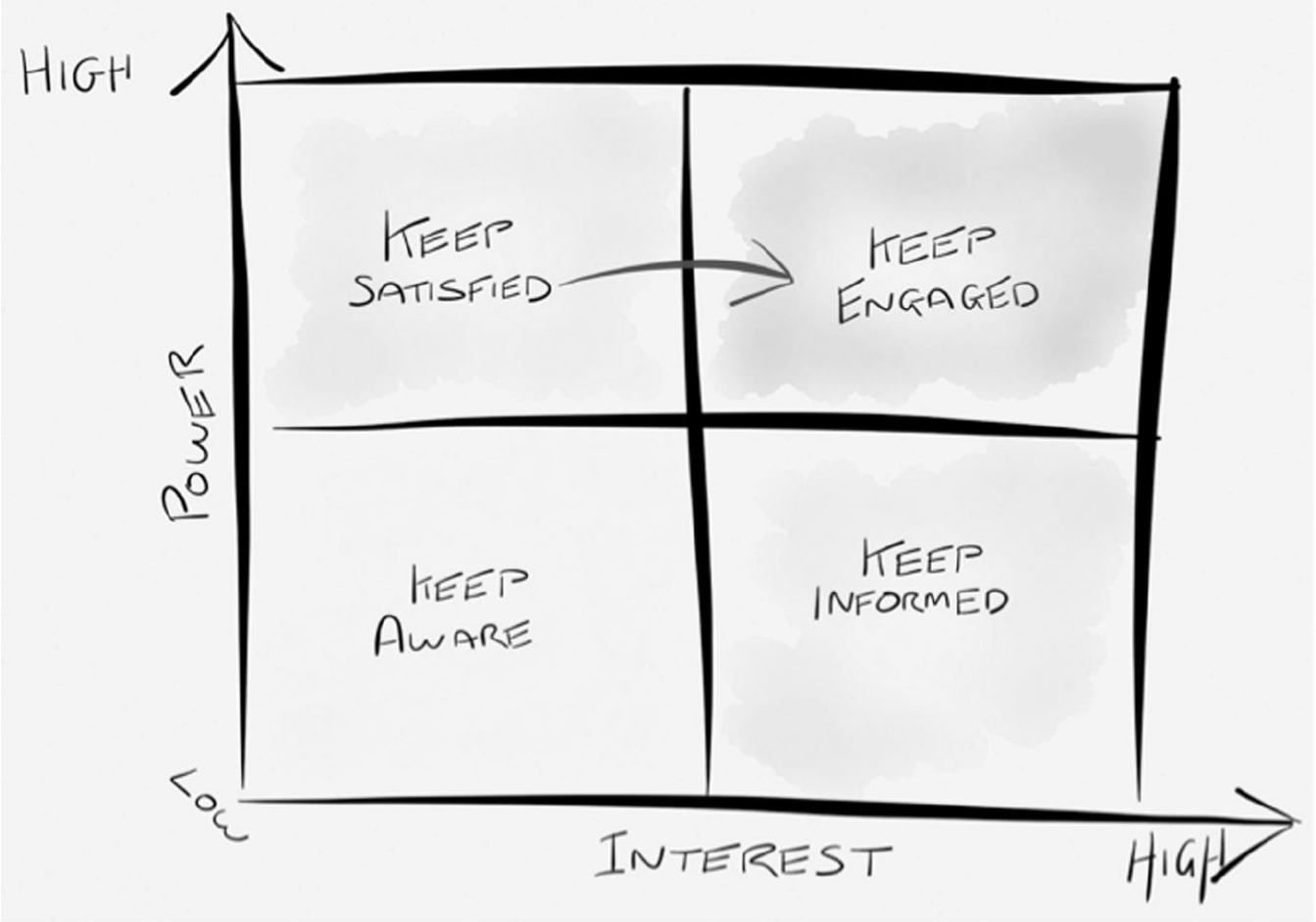
- Organisation charts and directories. Perhaps the first place to look for stakeholders is the company organisation chart or directory.
- Stakeholder lists. Creating a generic list of all stakeholders that may be involved in any product delivery is a good starting point to help new product deliveries identify potential stakeholders.
- Previous projects. Documentation from previous projects and talking to project teams can help to identify stakeholders that may have to be involved in a particular project type.
- Brainstorming and capturing every name, organisation or type of stakeholder that the participants in a workshop can think of is another great way of identifying stakeholders.

6.4.2 Stakeholder analysis

Once stakeholders have been identified, the next stage is to complete a stakeholder analysis. Different methodologies suggest different ways of analysing stakeholders, some complex and some very simple. The aim is to prioritise stakeholders in order of importance. A common approach is to map the interest and power or influence of each stakeholder group on a quadrant (Bryson, 2013; Figure 6.3).

- **High power/high interest** – these are key players and should be involved in governance and decision making. It is important to keep them engaged and to consult them regularly.
- **High power/low interest** – these should be consulted on their interest area with a view to increasing their level of interest. The aim is to move them into the high power/high interest quadrant. The danger with these stakeholders is that they perform ‘seagull management’ – they fly into the delivery intermittently, dump on everything, and then fly out again.
- **High interest/low power** – stakeholders in this group can be engaged by involving them in low-risk areas, keeping them informed and consulting on interest area. In fact the more creative and unexpected stories often come from the stakeholders in this bottom-right quadrant as this group are naturally interested in the product and often form an evangelist community of early adopters. The key with this community is to keep them interested and offer them early access to new stories (alpha community, focus groups, etc.).
- **Low interest/low power** – this group needs to be kept aware of what is going on via general communications: newsletters, websites, mailshots and so on.

Figure 6.3 Stakeholder power/interest mapping grid



Completing a power/interest grid will help to develop a communication plan that is aligned to each stakeholder’s focus and concerns.